3. Zeichen und Schrift

3.1 Medien Zeichen, Text, Schrift

3.2 Mikro-Typografie: Zeichensätze

3.3 Makro-Typografie: Gestalten mit Schrift

3.4 Hypertext und HTML
(Fortsetzung) 

- Allgemeines
- Textstrukturierung
- Tabellen
- Cascading Style Sheets 
- Framesets
- Medieneinbettung

Weitere Informationen: http://de.selfhtml.org/

Cascading Style Sheets (CSS)

- Von HTML prinzipiell unabhängige Sprache zur Beschreibung von Formatierungsinformation
  - Standardisierung durch W3C
  - Besonders für HTML geeignet
- Entstehungsgeschichte:
  - Vielzahl von "Standard-Attributen" in vielen HTML-Elementen (align, pos, color, font, ...)
  - Vereinheitlichung in CSS (aktuelle Version 2.0)
- In HTML 4.0 wird die Ablösung "alter" Konstrukte zugunsten einheitlicher CSS-beschriebener Styles forciert.
  - Universalattribut **`style`**
  - Alte Schreibweise (nicht mehr empfehlenswert):

    ```
    <p><font size="7">Text</font></p>
    ```

  - Neue Schreibweise mit CSS-Syntax:

    ```
    <p style="font-size:250%">Text</p>
    ```

CSS-Eigenschaften, Beispiel Schriftformatierung

- CSS-Syntax: Eigenschaft-Wert-Paare
  - Beispiel: **`font-size:250%`**
- Umfangreiche Liste an Eigenschaften und Maßeinheiten
- Eigenschaften zur Schriftformatierung:
  - **`font`** Zusammenfassung anderer Eigenschaften
  - **`font-family`** Gewünschte Schrift(en) mit Priorisierung
  - **`font-style`** Kursiv / normal
  - **`font-variant`** Kapitälchen *(small caps)* / normal
  - **`font-size`** Größe (numerisch oder ungenau)
  - **`font-weight`** Strichstärke (fett / mager)
  - **`font-stretch`** Laufweite
  - **`word-spacing`** Wortabstand
  - **`letter-spacing`** Zeichenabstand
  - **`color`** Farbe
  - **`...`**

CSS-Syntax

- Eigenschaft-Wert-Paar

  Eigenschaft **`:`** ***Wert*** z.B. **`font-style:italic`**

  – Wenn als Wert eines HTML-Attributs: Anführungszeichen "" empfehlenswert
- Mehrere Eigenschaft-Wert-Paare
  – Abtrennen mit Strichpunkt

    z.B. **`font-style:italic; font-size:large;`**
- Anführungszeichen für Werte (z.B. bei Leerzeichen im Wert)
  – Einfache Anführungszeichen ''

    z.B. **`font-family:'Times New Roman'`**
- Mehrere Werte (Sequenz) für eine Eigenschaft
  – Abtrennen mit Komma

    z.B. **`font-family:'Times New Roman', 'Times', serif`**

Weitere CSS-Eigenschaften

- Schriftformatierung (auch mit Schriftartendatei)
- Ausrichtung und Absatzkontrolle
- Außenrand und Abstand
- Innenabstand
- Rahmen
- Hintergrundfarben und -bilder
- Listenformatierung
- Tabellenformatierung
- Pseudoformate
  - z.B. **`link, visited, focus`**
- Positionierung und Anzeige von Elementen
- Layouts für Printmedien
- Sound-Kontrolle für Sprachausgabe
- Anzeigefenster

Einbindung von CSS in HTML (1)

- Individuell formatieren:
  - Universelles **`style`**-Attribut für alle HTML-Tags
  - z.B.
    ```
    <p style="font-weight:bold; font-size:200%">
    Beispieltext</p>
    ```
- Zentrale Stildefinitionen:
  - Festlegung der Style-Attribute für Standard-HTML-Elemente
  - z.B.

```
body {margin-left:100px; }
h1 { font-size:48pt;
     font-style:italic;
     border-bottom:solid thin black; }
p,li  { font-size:12pt;
        line-height:14pt;
        font-family:Helvetica,Arial,sans-serif;
        letter-spacing:0.2mm;
        word-spacing:0.8mm;
        color:blue; }
```

Einbindung von CSS in HTML (2)

- Ablage von zentralen Stildefinitionen im Kopfbereich der HTML-Datei

```
<style type="text/css">
... Stildefinitionen ...
</style>
```

  – Wegen Problemen älterer Browser oft Stildefinitionen als Kommentar

- Ablage von zentralen Stildefinitionen in separater CSS-Datei (.css)
  – Enthält nur Stildefinitionen, kein HTML
  – Einbindung in HTML-Dateien:

```
<link rel="stylesheet" type="text/css" href=Dateireferenz>
```

Beispiel zu CSS (Variante 1)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
   "http://www.w3.org/TR/html4/frameset.dtd">

<html>

  <head>
     <title>Beispiel zu CSS</title>
     <style>
          p     {font-family:Verdana; font-size:16pt}
          h1    {font-family:Verdana; color:green}
     </style>
  </head>

  <body>
     <h1>&Uuml;berschrift 1</h1>
     <p>Absatz 1</p>
     <h1>&Uuml;berschrift 2</h1>
     <p>Absatz 2</p>
     <h1>&Uuml;berschrift 3</h1>
     <p>Absatz 3</p>

  </body>
</html>
```

styles.html

Beispiel zu CSS (Variante 2)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
   "http://www.w3.org/TR/html4/frameset.dtd">

<html>

  <head>
     <title>Beispiel zu CSS</title>
     <link rel="stylesheet" type="text/css" href="styles.css">
  </head>

  <body>
     <h1>&Uuml;berschrift 1</h1>
     <p>Absatz 1</p>
     <h1>&Uuml;berschrift 2</h1>
     <p>Absatz 2</p>
     <h1>&Uuml;berschrift 3</h1>
     <p>Absatz 3</p>

  </body>
</html>
```

stylesfile.html

Datei **styles.css** (im gleichen Verzeichnis):

```
p     {font-family:Verdana; font-size:16pt}
h1    {font-family:Verdana; color:green}
```

Selbstdefinierte Stilklassen

- Eigene Stilklassen (außer den HTML-Elementen)
  - können frei definiert und verwandt werden
- Deklaration
  - bei der Stildefinition (mit dem Namen vorangestelltem Punkt)
  - z.B. `.navigation {font-size:16pt; color:blue;}`
- Anwendung
  - mit dem universellen `class`-Attribut aller HTML-Tags
  - z.B. `<li class="navigation">Home</li>`

Blockweise Formatierung mit CSS

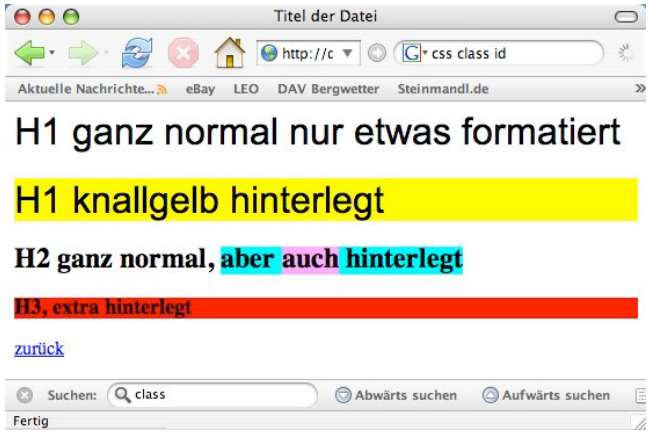
- Ganze Textbereiche einheitlich formatieren
- Verwendung des *Inline-Elements* **`<span> ... </span>`**
  - Keinerlei Effekt auf die Dokumentstruktur
  - Kann Text oder andere Inline-Elemente enthalten
  - Völlig äquivalent zur Wiederholung der angegebenen Stilangaben bei allen enthaltenen HTML-Elementen (mit **`style`**)
- Verwendung des *allgemeinen Blockelements* **`<div> ... </div>`**
  - Kann Text oder andere Blockelemente enthalten, z.B. auch Grafiken
  - Weitergabe der angegebenen Stilangaben zu allen enthaltenen HTML-Elementen
  - Kann mit der CSS-Eigenschaft **`position`** absolut positioniert werden
  - Kann mit Skripten ein- und ausgeblendet werden
  - Anmerkung: Oft benannt nach dem alten Netscape-spezifischen Element "layer"

Beispiel zu selbstdefinierten Stilklassen

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
   "http://www.w3.org/TR/html4/frameset.dtd">
<html>
  <head>
      <title>Beispiel zu CSS: Selbstdefinierte Klassen</title>
      <link rel="stylesheet" type="text/css" href="styles1.css">
  </head>
  <body>
      <p>Dies ist ein ganz normaler Absatz ohne spezielle
      Formatierung. </p>
      <p class="merksatz">Dies ist ein Merksatz,
      speziell formatiert mit Hilfe von CSS.</p>
      <p>Dies ist wieder ein ganz normaler Absatz.</p>
      <span class="programm">
            <p>Dies sind zwei aufeinander folgende Abs&auml;tze,
            die speziell formatiert werden.</p>
            <p>Dies ist der zweite solche Absatz.</p>
      </span>
  </body>
</html>
```

stylesclass.html

Klassen: Komplexeres Beispiel




```
<html>
<head>
<title>Titel der Datei</title>
<style type="text/css">
h1 {font-family:Arial,sans-serif; font-size:2em; font-weight:normal;}
h1.hinterlegt { background-color:yellow }
*.hinterlegt { background-color:cyan}
.extra { background-color:magenta}
.extra.hinterlegt { background-color:red}
</style>
</head>
<body>
<h1>H1 ganz normal nur etwas formatiert</h1>
<h1 class="hinterlegt">H1 knallgelb hinterlegt</h1>
<h2>H2 ganz normal, <span class="hinterlegt"> aber <b
  class="extra">auch</b> hinterlegt</span></h2>
<h3 class="extra hinterlegt">H3, extra hinterlegt</h3>
</body>
</html>
```

stylescomplex.html

ID statt class?

- Im HTML file: <h1 ID="blau">Überschrift </h1>
- Im CSS file: #blau {color : blue;}

- Die Benutzung einer ID bietet einige Vorteile:
  - Sie kann als Sprungziel für Hyperlinks verwendet werden.
  - IDs können mit Javascript angesprochen werden. Das funktioniert über getElementById()
  - IDs überstimmen Klassen.
- Nachteile von IDs:
  - Eine ID darf auf einer Seite nur einmal verwendet werden. Auch wenn so mancher Browser das anders sieht.
  - IDs können nicht wie Klassen kombiniert werden.

- Klassen und IDs können gemeinsam genutzt werden. Beispiel:

  <h1 class="font" ID="blau">Überschrift </h1>

3. Zeichen und Schrift

3.1 Medien Zeichen, Text, Schrift

3.2 Mikro-Typografie: Zeichensätze

3.3 Makro-Typografie: Gestalten mit Schrift

3.4 Hypertext und HTML
(Fortsetzung)

- Allgemeines
- Textstrukturierung
- Cascading Style Sheets
- Tabellen
- Framesets
- Medieneinbettung

Weitere Informationen: http://selfhtml.teamone.de

Framesets

- Einteilung einer Seite in separate Segmente *(frames)*
  - Die Gesamtseite definiert ein so genanntes *frameset.*
  - Jedes Einzelframe liegt in einer Einzeldatei.
  - Anzeige der Frames ist unabhängig voneinander (werden separat geladen).

- Grundgerüst eines Frameset:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
   "http://www.w3.org/TR/html4/frameset.dtd">
<html>
  <head>
    <title>Text des Titels</title>
  </head>
  <frameset ...>  <!-- Frameset-Definition -->
    <frame ...>   <!-- Framefenster-Definition -->
    <noframes>
      Wird angezeigt, wenn der Browser keine Frames anzeigen kann
    </noframes>
  </frameset>
</html>
```

Beispiel: Basisdatei eines Framesets

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
  "http://www.w3.org/TR/html4/frameset.dtd">
<html>
<head>
<title>Beispiel zu Frames</title>
</head>
<frameset cols="250,*">
  <frame src="verweise.html" name="Navigation">
  <frame src="startseite.html" name="Daten">
  <noframes>
    Ihr Browser kann diese Seite leider nicht anzeigen!
  </noframes>
</frameset>
</html>
```

Aufteilung der Seite:

- Horizontal (rows) oder vertikal (cols)
- In absoluten (Pixel-)Zahlen oder prozentual
- "*" = Rest der Fläche
- Komma-getrennte Liste

Anzeige von Inhalten in Framesets

- Bei Verweisen kann mit dem **`target`**-Attribut festgelegt werden, in welchem Frame die Anzeige erfolgt.

Beispiel: Datei "verweise.html"

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
       "http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <title>Text des Titels</title>
  </head>
  <body>
   <h1>Navigation</h1>
   <p>
    <a href="allgem.html" target="Daten"><b>Allgemeines</b></a><br>
    <a href="styles.html" target="Daten"><b>Styles</b></a><br>
    <a href="table.html" target="Daten"><b>Tabellen</b></a>
   </p>
  </body>
</html>
```

Verwendung des "target"-Attributs

- Werte für "**target**"-Attribut in Links (Anchor-Tag **<a>**):
  - **_blank** = Verweis in neuem Fenster öffnen
  - **_self** = Verweis im gleichen Fenster öffnen
  - **_parent** = aktuelles Frameset beim Ausführen des Verweises sprengen
  - **_top** = alle Framesets beim Ausführen des Verweises sprengen
- "target" auch außerhalb von Framesets anwendbar
  - **_blank**
- Beispiel:

```
Hier ist ein
<a target="_blank" href="allgem.html">Link</a>,
der ein neues Fenster öffnet.
```

- Moderne Browser öffnen oft einen neuen „Tab“ statt eines Fensters.

newwindow.html

Vor- und Nachteile von Framesets

- Vorteile:
  - Elegante Gestaltungsmöglichkeiten
  - Navigationshilfen bleiben auch beim Blättern in angezeigter Information am gleichen Platz
  - Nachladen von Einzelinformation u.U. schneller als ohne Frames
  - Parallele Anzeige von Information z.B. zu Vergleichszwecken
- Nachteile:
  - Inkompatibilität mit älteren Browsern
  - Gesamtladezeit schlechter als ohne Frames
  - Einzelansichten nicht mehr als Ganzes adressierbar
    - » Bruch mit den Grundparadigmen von HTML?
  - Suchmaschinen indizieren oft Teile von Framesets
- Empfehlungen:
  - Frames nur da einsetzen, wo wirklich sinnvoll!
  - Idealerweise Frame-freie Alternative (nicht nur Fehlertext) anbieten

Iframe: Eingebettete Frames

- Möglichkeit, Frames innerhalb einer normalen HTML Datei zu verwenden
- Verhalten wie eingebettete Grafik: Scrollt mit dem Text
- Eigener Rollbalken bei großen Inhalten
- Anwendung z.B. für Werbungseinblendung
- Vielfältige Gestaltungsmöglichkeiten, aber Frame-spezifische Probleme bleiben bestehen

3. Zeichen und Schrift

3.1 Medien Zeichen, Text, Schrift

3.2 Mikro-Typografie: Zeichensätze

3.3 Makro-Typografie: Gestalten mit Schrift

3.4 Hypertext und HTML
(Fortsetzung)

- Allgemeines
- Textstrukturierung
- Cascading Style Sheets
- Tabellen
- Framesets
- Medieneinbettung

Weitere Informationen: http://selfhtml.teamone.de

Integration von Bildern

- Bilder einbinden mit **<img>**
- Attribut **src** gibt Quelle an (auch von anderen Servern möglich)
  - Achtung Copyright-Fragen!
- Größenangaben mit **width** und **height**
  - Bei Angabe beider Werte Verzerrung möglich
- Bilder können auch als Inhalt eines Verweises vorkommen
  - z.B. grafische Navigationsleisten

```
<html> ...
 <body>
   <h1>Ein JPEG-Bild des Eiffelturms</h1>

   <p><img src="Eiffelturm.jpg" width="350" alt="Eiffelturm">
   </p>

 </body>
</html>
```

Integration anderer Dateien

- Prinzipiell alle Dateien einbettbar
  - mit dem **<object>**-Tag
  - als Hyperlinks
- Beispiel zu Sound:
  - Achtung: nur in neueren Browsern unterstützt
  - Ältere Variante: **<embed>**-Tag

```
<p>Sound-Objekt
      <object data="../sounds/bgndmusic.mid" type="audio/midi">
      Ihr Browser kann das Objekt leider nicht anzeigen!
      </object>
</p>

<p>Sound als Link<br>
      <a href="../sounds/technobop.mid"
            type="audio/midi">Bitte klicken!</a>
</p>
```

MIME

- MIME = Multipurpose Internet Mail Extensions
  - In HTML mit dem **`type`**-Attribut an vielen Stellen angebbar (z.B. **`<link>, <object>`**)
  - Erleichtert dem Browser (bzw. seinem Benutzer) die Entscheidung, wie Dateien zu behandeln sind
  - Jeder Browser führt eine Liste der akzeptierten MIME-Extensions und Regeln für die Behandlung (z.B. speichern, Programm aufrufen)
  - Liste siehe http://www.iana.org/assignments/media-types

- Syntax:

  `Medientyp / Untertyp`

  - Medientypen: text, image, video, audio, application, ...
  - Untertypen, die auf dem Server auszuführen sind, beginnen meist mit x-
  - Hersteller- *(vendor-)*spezifische Untertypen im speziellen Unterbaum "vnd."

Design vs. Flexibilität

- Aus gestalterischer Motivation werden oft folgende Konzepte verwendet:
  - Feste Formatvorgaben für die Seite
  - Spezial-Schriften
  - Feste Schriftgrößen
  - Frames
  - Aufwändige grafische Elemente
- Die maximale Flexibilität in der Verwendung spricht für:
  - Flexible Fenstergröße („liquid design“)
  - Unabhängigkeit von Schriftwahl
  - Vom Benutzer bestimmbare Schriftgrößen
  - Keine oder sehr eingeschränkte Benutzung von Frames
  - Kleine, sparsame grafische Elemente

Barrierefreiheit von Webseiten

- Gesellschaftliche Funktionen des WWW:
  - Wesentliches Medium für staatliche Informationsdienste und Bürgerservice
  - Tendenziell besonders leicht zugänglich für Personen, die andere Zugänge nur schwer nutzen können (z.B. Behinderte)
  - Generell ein demokratisches Medium, das für alle offen sein soll
- Nutzung des WWW bei eingeschränkten Wahrnehmungs- und Aktionsmöglichkeiten
  - Seh- oder Hörbehinderung
  - Leseschwäche, Aufnahme-, Lernschwäche
  - Einschränkungen bei der Benutzung von „zeigenden“ Eingabegeräten
- Richtlinien: Web Accessibility Initiative (http://www.w3.org/WAI/), Beispiele:
  - Ergänzung grafischer Information durch textuelle Beschreibung
    - » Auch bei zeitabhängigen Medien (Untertitel zu Video)
  - Benutzbarkeit mit Tastatur (d.h. auch mit Spracheingabe)
  - Orientierung durch klare Struktur und kleine Textblöcke erleichtern
  - Hoher Kontrast zwischen Vordergrund und Hintergrund
  - Auslösung epileptischer Anfälle durch blinkende Inhalte verhindern